LAPORAN PRAKTIKUM
PEMROGRAMAN MOBILE
MODUL 2

ANDROID LAYOUT WITH COMPOSE

Oleh:

Muhammad Rizki Ramadhan NIM. 2310817310008

PROGRAM STUDI TEKNOLOGI INFORMASI
FAKULTAS TEKNIK
UNIVERSITAS LAMBUNG MANGKURAT
APRIL 2025

LEMBAR PENGESAHAN
LAPORAN PRAKTIKUM PEMROGRAMAN I
MODUL 2

Laporan Praktikum Pemrograman Mobile Modul 2: Android Layout With Compose ini disusun sebagai syarat lulus mata kuliah Praktikum Pemrograman Mobile. Laporan Prakitkum ini dikerjakan oleh:

Nama Praktikan : Muhammad Rizki Ramadhan
NIM : 2310817310008

Menyetujui, Asisten Praktikum	Mengetahui, Dosen Penanggung Jawab Praktikum
Muhammad Raka Azwar NIM. 2210817210012	Andreyan Rizky Baskara, S.Kom., M.Kom. NIP. 19930703 20190301011

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

SOAL 1

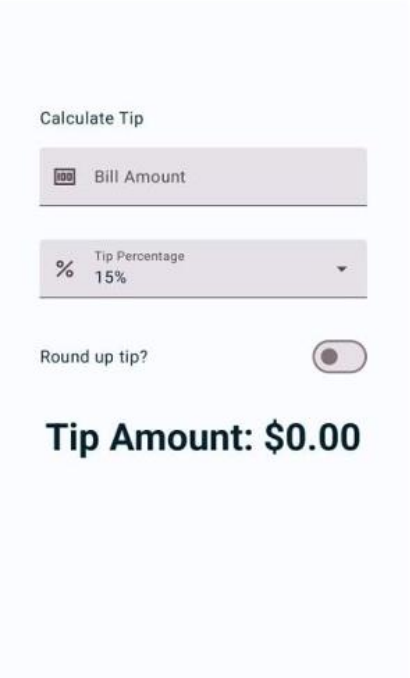
Buatlah sebuah aplikasi kalkulator tip menggunakan XML dan Jetpack Compose yang dirancang untuk membantu pengguna menghitung tip yang sesuai berdasarkan total biaya layanan yang mereka terima. Fitur-fitur yang diharapkan dalam aplikasi ini mencakup:

a. Input biaya layanan: Pengguna dapat memasukkan total biaya layanan yang diterima dalam bentuk nominal.

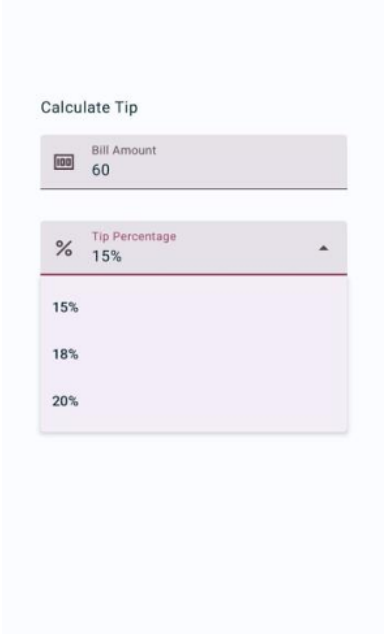
b. Pilihan persentase tip: Pengguna dapat memilih persentase tip yang diinginkan.

c. Pengaturan pembulatan tip: Pengguna dapat memilih untuk membulatkan tip ke angka yang lebih tinggi.

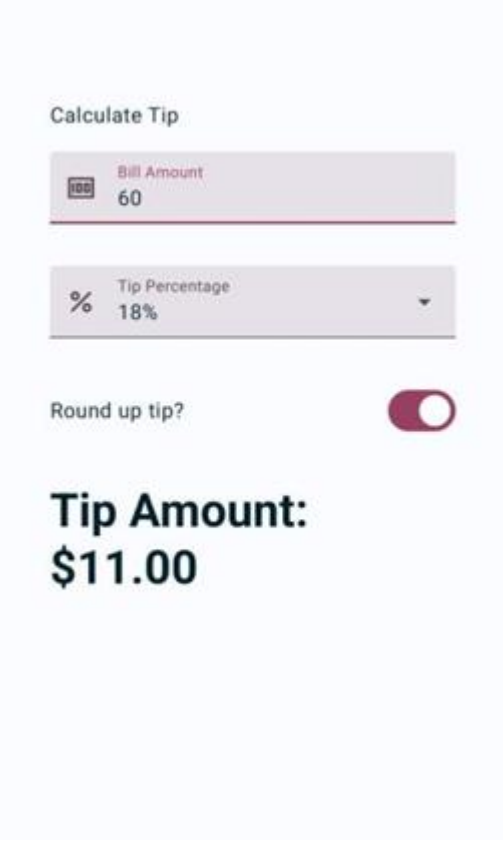
d. Tampilan hasil: Aplikasi akan menampilkan jumlah tip yang harus dibayar secara langsung setelah pengguna memberikan input.



Gambar 1. Tampilan Awal Aplikasi



Gambar 2. Tampilan Dadu Setelah Diroll



Gambar 3. Tampilan Roll Dadu Double

A. Source Code

XML:

MainActivity.kt

```
package com.example.tipcalculatorxml

import android.icu.text.NumberFormat
import android.os.Bundle
import android.widget.ArrayAdapter
import android.widget.AutoCompleteTextView
import androidx.appcompat.app.AppCompatActivity
import
com.example.tipcalculatorxml.databinding.ActivityMainBinding
import androidx.core.widget.addTextChangedListener
import java.util.Locale
import kotlin.math.ceil


class MainActivity : AppCompatActivity() {

    private lateinit var binding: ActivityMainBinding
    private val tipPercentage = listOf ("15%", "18%", "20%")
    private var selectedTipPercentage = 15.0
    private var roundUp = false

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        binding = ActivityMainBinding.inflate(layoutInflater)
        setContentView(binding.root)

        setupTipPercentageInput()
        setupBillInput()
        setupRoundUpTip()
    }

    private fun setupTipPercentageInput() {
        val adapter = ArrayAdapter(this,
android.R.layout.simple_dropdown_item_1line, tipPercentage)
        binding.tipPercentageInput.setAdapter(adapter)

        binding.tipPercentageInput.setOnFocusChangeListener {
```

```
view,                          hasFocus                          ->
            if                     (hasFocus)                     {
                (view as? AutoCompleteTextView)?.showDropDown()
            }
        }

        binding.tipPercentageInput.setOnItemClickListener { _,
_,              position,                  _                      ->
            selectedTipPercentage                                 =
tipPercentage[position].dropLast(1).toDouble()
            val                    billAmount                     =
binding.billAmountInput.text.toString().toDoubleOrNull() ?: 0.0
            updateCalculatedTip(billAmount)
        }
    }



            private             fun             setupBillInput(){
        binding.billAmountInput.addTextChangedListener{editable
->
            val                    billAmount                     =
editable?.toString()?.toDoubleOrNull()          ?:          0.0
            updateCalculatedTip(billAmount)
        }
    }

    private                fun                setupRoundUpTip(){
        binding.roundUpSwitch.setOnCheckedChangeListener    {_,
isChecked                                                         ->
            roundUp                  =                  isChecked
            val                    billAmount                     =
binding.billAmountInput.text.toString().toDoubleOrNull() ?: 0.0
            updateCalculatedTip(billAmount)
        }
    }

    private   fun   updateCalculatedTip(billAmount:   Double){
        var tip = selectedTipPercentage / 100 * billAmount
        if                        (roundUp)                       {
            tip                     =                   ceil(tip)
        }
        val                     formattedTip                      =
NumberFormat.getCurrencyInstance(Locale.US).format(tip)
        binding.calculatedTip.text                                =
getString(R.string.tip_amount)   +   "   "   +   formattedTip
```

```
    }
}
```

Table 1. Source Code Jawaban Soal 1 XML

activity_main.xml

```
<?xml              version="1.0"              encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/title_text"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="40dp"
        android:text="@string/calculate_tip"
        android:textSize="20sp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.109"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.034"           />

    <com.google.android.material.textfield.TextInputLayout
        android:id="@+id/bill_amount_layout"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginTop="20dp"
        android:layout_marginStart="32dp"
        android:layout_marginEnd="32dp"
        app:layout_constraintTop_toBottomOf="@+id/title_text"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:startIconDrawable="@drawable/money">
```

```
<com.google.android.material.textfield.TextInputEditText
            android:id="@+id/billAmountInput"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:hint="@string/bill_amount"
            android:importantForAccessibility="yes"
            android:paddingStart="50dp"
            android:inputType="numberDecimal"                />
    </com.google.android.material.textfield.TextInputLayout>

    <com.google.android.material.textfield.TextInputLayout
        android:id="@+id/tip_percentage_layout"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginTop="20dp"
        android:layout_marginStart="32dp"
        android:layout_marginEnd="32dp"
        android:hint="@string/tip_option"
        app:startIconDrawable="@drawable/baseline_percent_24"

app:layout_constraintTop_toBottomOf="@id/bill_amount_layout"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent">

        <AutoCompleteTextView
            android:id="@+id/tipPercentageInput"
            android:layout_width="match_parent"
            android:layout_height="55dp"
            android:inputType="none"
            android:focusable="true"
            android:cursorVisible="false"
            android:clickable="true"
            android:dropDownHeight="wrap_content"
            android:dropDownVerticalOffset="10dp"
            android:importantForAccessibility="yes"
            android:paddingStart="50dp"                      />
    </com.google.android.material.textfield.TextInputLayout>
    <LinearLayout
        android:id="@+id/round_up_row"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:orientation="horizontal"

app:layout_constraintTop_toBottomOf="@id/tip_percentage_layout"
        app:layout_constraintStart_toStartOf="parent"
```

```
        app:layout_constraintEnd_toEndOf="parent"
        android:layout_marginTop="32dp"
        android:paddingStart="16dp"
        android:paddingEnd="16dp"
        android:gravity="center_vertical">

        <TextView
            android:id="@+id/round_up_text"
            android:layout_width="139dp"
            android:layout_height="match_parent"
            android:layout_gravity="start"
            android:layout_marginStart="15dp"
            android:text="@string/round_up_tip"
            android:textSize="18sp"                          />

        <Switch
            android:id="@+id/roundUpSwitch"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:minWidth="48dp"
            android:minHeight="48dp"
            android:layout_marginStart="150dp"
            android:importantForAccessibility="yes"          />
    </LinearLayout>

    <TextView
        android:id="@+id/calculated_tip"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="36dp"
        android:text="@string/tip_amount"
        android:textSize="25sp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.181"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@id/round_up_row"
/>

</androidx.constraintlayout.widget.ConstraintLayout>
```

Table 2. Source Code Jawaban Soal 1 XML

Jetpack Compose:

MainActivity.kt

```
package com.example.tipcalculatorcompose

import androidx.compose.material3.ExperimentalMaterial3Api
import android.os.Bundle
import androidx.activity.ComponentActivity
import androidx.activity.compose.setContent
import androidx.activity.enableEdgeToEdge
import androidx.compose.material.icons.Icons
import androidx.annotation.StringRes
import androidx.compose.foundation.layout.Arrangement
import androidx.compose.foundation.layout.Column
import androidx.compose.foundation.layout.Row
import androidx.compose.foundation.layout.Spacer
import androidx.compose.foundation.layout.fillMaxSize
import androidx.compose.foundation.layout.fillMaxWidth
import androidx.compose.foundation.layout.height
import androidx.compose.foundation.layout.padding
import androidx.compose.foundation.rememberScrollState
import androidx.compose.foundation.text.KeyboardOptions
import androidx.compose.foundation.verticalScroll
import androidx.compose.material.icons.filled.Money
import androidx.compose.material.icons.filled.Percent
import androidx.compose.material3.DropdownMenuItem
import androidx.compose.material3.ExposedDropdownMenuBox
import androidx.compose.material3.ExposedDropdownMenuDefaults
import androidx.compose.material3.Icon
import androidx.compose.material3.MaterialTheme
import androidx.compose.material3.Surface
import androidx.compose.material3.Switch
import androidx.compose.material3.Text
import androidx.compose.material3.TextField
import androidx.compose.runtime.Composable
import androidx.compose.runtime.getValue
import androidx.compose.runtime.mutableStateOf
import androidx.compose.runtime.remember
import androidx.compose.runtime.setValue
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.res.stringResource
import androidx.compose.ui.text.input.ImeAction
import androidx.compose.ui.text.input.KeyboardType
import androidx.compose.ui.tooling.preview.Preview
import androidx.compose.ui.unit.dp
```

```
import
com.example.tipcalculatorcompose.ui.theme.TipCalculatorComposeTheme
import java.text.NumberFormat

class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContent {
            TipCalculatorComposeTheme {
                Surface (
                    modifier = Modifier.fillMaxSize(),
                ) {
                    TipCalculatorLayout()
                }
            }
        }
    }
}
@OptIn(ExperimentalMaterial3Api::class)
@Composable
fun TipCalculatorLayout() {
    var amountInput by remember { mutableStateOf("") }
    var calculatedTip by remember { mutableStateOf("") }
    var roundUp by remember { mutableStateOf(false) }
    var expanded by remember { mutableStateOf(false) }

    val optionsTipPercentage = listOf("15%", "18%", "20%")
    var         selectedOption         by         remember         {
mutableStateOf(optionsTipPercentage[0]) }

    Column(
        modifier = Modifier
            .padding(horizontal = 40.dp)
            .verticalScroll(rememberScrollState()),
        horizontalAlignment = Alignment.CenterHorizontally,
        verticalArrangement = Arrangement.Center
    ) {
        Text(
            text = stringResource(R.string.calculate_tip),
            modifier = Modifier
                .padding(bottom = 16.dp, top = 40.dp)
                .align(alignment = Alignment.Start)
        )
```

```
        EditNumberField(
            label = R.string.bill_amount,
            value = amountInput,
            onValueChanged = {
                amountInput = it
                calculatedTip        =        calculateTipInput(it,
selectedOption, roundUp)
                            },
            leadingIcon      =      {      Icon(Icons.Filled.Money,
contentDescription = null) },
            keyboardOptions = KeyboardOptions(
                keyboardType = KeyboardType.Number,
                imeAction = ImeAction.Next
            ),
            modifier = Modifier
                .padding(bottom = 32.dp)
                .fillMaxWidth()
        )

        ExposedDropdownMenuBox(
            expanded = expanded,
            onExpandedChange = { expanded = !expanded }
        ) {
            TextField(
                readOnly = true,
                value = selectedOption,
                onValueChange = {},
                label = { Text(stringResource(R.string.tip_option))
},
                leadingIcon    =    {    Icon(Icons.Default.Percent,
contentDescription = null)},
                trailingIcon = {

ExposedDropdownMenuDefaults.TrailingIcon(expanded = expanded)
                },
                colors                                                =
ExposedDropdownMenuDefaults.textFieldColors(),
                modifier = Modifier.menuAnchor().fillMaxWidth()
            )

            ExposedDropdownMenu(
                expanded = expanded,
                onDismissRequest = { expanded = false }
            ) {
                optionsTipPercentage.forEach { selectionOption ->
```

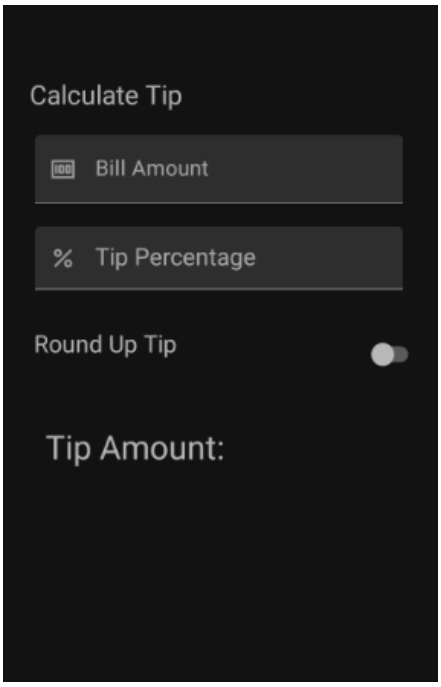
```
                    DropdownMenuItem(
                        text = { Text(text = selectionOption) },
                        onClick = {
                            selectedOption = selectionOption
                            expanded = false
                            calculatedTip                          =
calculateTipInput(amountInput, selectedOption, roundUp)
                        }
                    )
                }
            }
        }

        RoundTheTipRow(
            roundUp = roundUp,
            onRoundUpChanged = {
                roundUp = it
                calculatedTip    =    calculateTipInput(amountInput,
selectedOption, it)
            },
            modifier = Modifier.padding(bottom = 32.dp)
        )

        Text(
            text        =        stringResource(R.string.tip_amount,
calculatedTip),
            style = MaterialTheme.typography.displaySmall
        )

        Spacer(modifier = Modifier.height(150.dp))
    }
}

private fun calculateTipInput(amountInput: String, selectedTipText:
String, roundUp: Boolean): String {
    val amount = amountInput.toDoubleOrNull() ?: 0.0
    val tipPercent = selectedTipText.dropLast(1).toDoubleOrNull() ?:
15.0
    return calculateTip(amount, tipPercent, roundUp)
}

@Composable
fun EditNumberField(
    value: String,
    @StringRes label: Int,
```

```
    leadingIcon: @Composable (() -> Unit)? = null,
    onValueChanged: (String) -> Unit,
    keyboardOptions: KeyboardOptions,
    modifier: Modifier = Modifier
) {

    TextField(
        value = value,
        singleLine = true,
        modifier = modifier,
        onValueChange = onValueChanged,
        label = { Text(stringResource(label)) },
        leadingIcon = leadingIcon,

        keyboardOptions = keyboardOptions
    )
}

@Composable
fun RoundTheTipRow(
    roundUp: Boolean,
    onRoundUpChanged: (Boolean) -> Unit,
    modifier: Modifier = Modifier
) {
    Row(
        modifier = modifier
            .fillMaxWidth()
            .height(48.dp),
        verticalAlignment = Alignment.CenterVertically,
        horizontalArrangement = Arrangement.SpaceBetween
    ) {
        Text(
            text = stringResource(R.string.round_up_tip),
            modifier = Modifier.weight(1f)
        )
        Switch(
            checked = roundUp,
            onCheckedChange = onRoundUpChanged,
        )
    }
}

private fun calculateTip(amount: Double, tipPercent: Double = 15.0,
roundUp: Boolean): String{
    var tip = tipPercent / 100 * amount
```

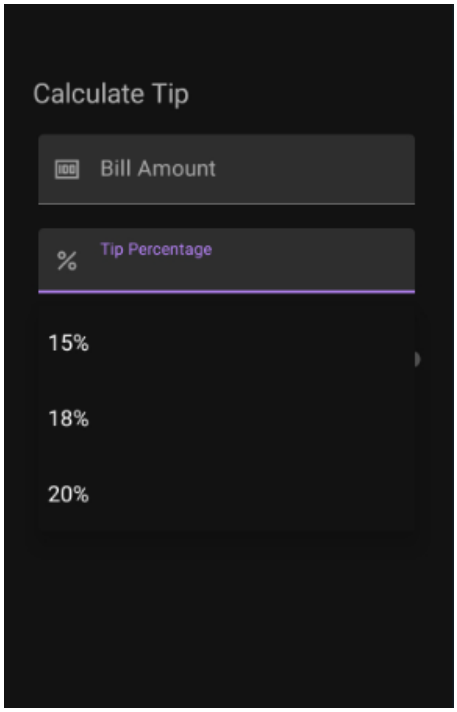
```
    if (roundUp){
        tip = kotlin.math.ceil(tip)
    }

    return
NumberFormat.getCurrencyInstance(java.util.Locale.US).format(tip)
}

@Preview(showBackground = true)
@Composable
fun TipCalculatorLayoutPreview(){
    TipCalculatorComposeTheme{
        TipCalculatorLayout()
    }
}
```

Table 3. Source Code Jawaban Soal 1 Jetpack Compose

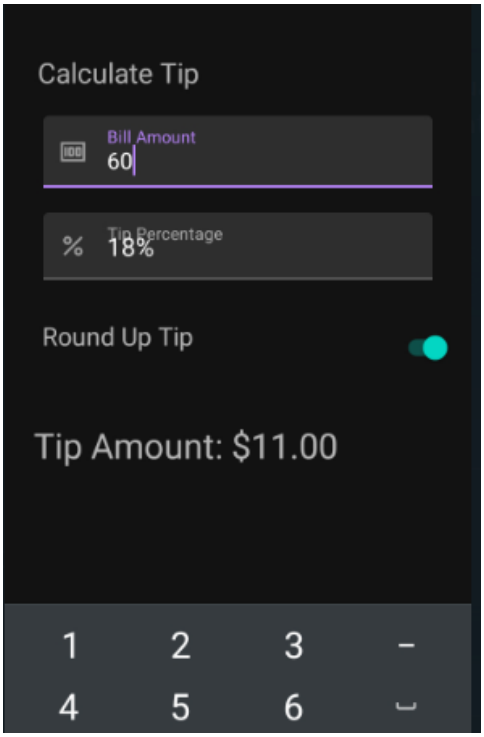
B. Output Program



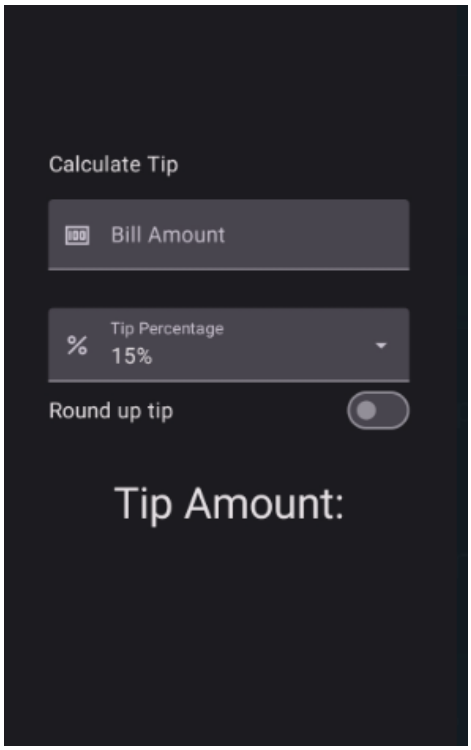
Gambar 4. Tampilan Awal Aplikasi XML



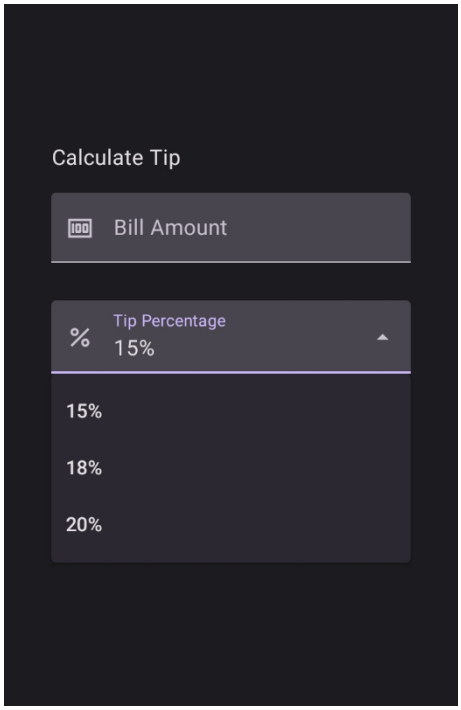
Gambar 5. Tampilan Pilihan Persentase Tip XML



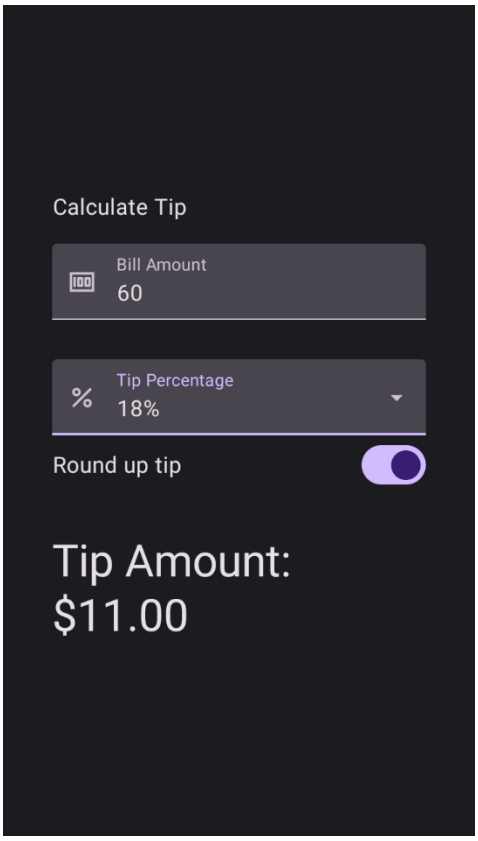
Gambar 6. Tampilan Aplikasi Setelah Dijalankan XML



Gambar 7. Tampilan Awal Aplikasi Jetpack Compose



Gambar 8. Tampilan Pilihan Persentase Tip Jetpack Compose



Gambar 9. Tampilan Aplikasi Setelah Dijalankan Jetpack Compose

C. Pembahasan

MainActivity.kt (XML):

1. Pada line [1], syntax package `package com.example.tipcalculatorxml` merupakan deklarasi nama package file Kotlin.
2. Pada line [3] sampai line [11], `import android.icu.text.NumberFormat, import android.os.Bundle, import android.widget.ArrayAdapter, import android.widget.AutoCompleteTextView, import androidx.appcompat.app.AppCompatActivity, import com.example.tipcalculatorxml.databinding.ActivityMainBinding, import androidx.core.widget.addTextChangedListener, import java.util.Locale, import kotlin.math.ceil` merupakan program import class dan library yang diperlukan, seperti format mata uang, menangani

lifecycle onCreate, dropdown, base class activity, viewbinding, perubahan teks, format lokal, dan pembulatan ke atas.

3. Pada line [14], syntax `class MainActivity : AppCompatActivity() {}` merupakan deklarasi kelas bernama MainActivity yang extend dari AppCompatActivity.
4. Pada line [16], `private lateinit var binding: ActivityMainBinding` mendeklarasikan objek binding untuk mengakses view dari XML via viewBinding.
5. Pada line [17], `private val tipPercentage = listOf ("15%", "18%", "20%")` merupakan list opsi persentase tip dalam bentuk string.
6. Pada line [18], `private var selectedTipPercentage = 15.0 private var roundUp = false` merupakan program untuk menyimpan persentase tip terpilih dan status pembulatan tip.
7. Pada line [21] sampai line [29], `override fun oncreate` program untuk menginisialisasi binding, menampilkan UI, dan setup fungsi input.
8. Pada line [31] sampai line [48], `private fun setupTipPercentageInput() {` merupakan fungsi untuk menangani dropdown tip.
9. Pada line [33] sampai line [34], `val adapter = ArrayAdapter(this, android.R.layout.simple_dropdown_item_1line,tipPercentage) binding.tipPercentageInput.setAdapter(adapter)` membuat dan menetapkan adapter ke AutoCompleteTextView agar tampil sebagai dropdown.
10. Pada line [43] sampai line [48], `binding.tipPercentageInput.setOnItemClickListener { _, _, position, _ -> selectedTipPercentage = tipPercentage[position].dropLast(1).toDouble() val billAmount = binding.billAmountInput.text.toString().toDoubleOrNull() ?: 0.0 updateCalculatedTip(billAmount)} }` berfungsi saat user memilih item, simpan nilai persentase, ambil bill saat ini, lalu update hasil tip.
11. Pada line [51] sampai line [56], `private fun setupBillInput(){ binding.billAmountInput.addTextChangedListener{editable -> val billAmount = editable?.toString()?.toDoubleOrNull() ?: 0.0 updateCalculatedTip(billAmount) }}` berfungsi saat user mengetik angka pada bill, ambil nilainya dan update hasil tip.
12. Pada line [58] sampai line [64], `private fun setupRoundUpTip(){ binding.roundUpSwitch.setOnCheckedChangeListener {_, isChecked -> roundUp = isChecked val billAmount =`

`binding.billAmountInput.text.toString().toDoubleOrNull() ?: 0.0 updateCalculatedTip(billAmount)}}` berfungsi saat switch round-up diubah, simpan statusnya dan hitung ulang tip.

13. Pada line [66] sampai line [72], `private fun updateCalculatedTip(billAmount: Double){ var tip = selectedTipPercentage / 100 * billAmount if (roundUp) { tip = ceil(tip)} val formattedTip = NumberFormat.getCurrencyInstance(Locale.US).format(tip) binding.calculatedTip.text = getString(R.string.tip_amount) + " " + formattedTip}` berfungsi untuk menghitung tip, mengalikan bill dengan persentase, jika perlu dibulatkan, gunakan ceil(), dan memformat ke mata uang dan tampilkan ke UI.

activity_main.xml:

1. Pada line [1], syntax <?xml version="1.0" encoding="utf-8"?> sebagai deklarasi XML.
2. Pada line [2], <androidx.constraintlayout.widget.ConstraintLayout merupakan Layout utama menggunakan ConstraintLayout, memungkinkan penempatan komponen berdasarkan hubungan antar elemen.
3. Pada line [5] sampai line [8], digunakan untuk ID dari layout utama, ukuran layout memenuhi seluruh layar (match_parent), dan hanya untuk Android Studio agar preview tahu class aktivitas.
4. Pada line [10] sampai line [22], syntax berfungsiuntuk judul layar: **"Calculate Tip"**, memberi ruang dari atas, ukuran teks sedang, menyematkan posisi ke atas, kiri, kanan.
5. Pada line [24], syntax berfungsi untuk membungkus untuk TextInputEditText, memberikan fitur Material seperti ikon, label, animasi input.
6. Pada line [36] sampai line [43], syntax berfungsi untuk membuat kolom input angka desimal untuk jumlah tagihan, hint tampil saat input kosong, dan memunculkan keyboard angka dengan titik desimal.
7. Pada line [47] sampai line [55], sama seperti sebelumnya, tapi untuk dropdown tip dan menambahkan ikon persen di sisi kiri.
   Pada line [60] sampai line [72], AutoCompleteTextView digunakan untuk memilih dari persentase tip yang tersedia, agar terlihat seperti dropdown, bukan input biasa.
8. Pada line [75] sampai line [86], berfungsi sebagai baris horizontal yang berisi label dan Switch.
9. Pada line [88] sampai line [95], berfungsi untuk membuat label: "Round Up Tip".
10. Pada line [99] sampai line [106],berfungsi untuk membuat tombol Switch untuk memilih apakah tip dibulatkan.

11. Pada line [111] sampai line [121], berfungsi untuk menampilkan hasil kalkulasi tip, misalnya: Tip Amount: $3.45.
12. Pada line [123], menutup root layout constraintLayout.

MainActivity.kt (Jetpack Compose):

1. Pada line [1], syntax package `package com.example.tipcalculatorcompose` merupakan deklarasi nama package file Kotlin.
2. Pada line [3] sampai line [45], merupakan program import class dan library yang diperlukan, seperti layout, material icons, state management, dan sebagainya.
3. Pada line [47] sampai line [61], syntax `class MainActivity : ComponenttActivity() {}` merupakan activity untuk Jetpack Compose yang berisi program UI penuh di layer, mengatur UI dengan compose, dan program untuk tema dan memanggil composable utama.
4. Pada line [48], `override fun onCreate(savedInstanceState: Bundle?) {` merupakan override fungsi onCreate untuk menginisialisasi UI saat activity dibuat.
5. Pada line [50], `enableEdgeToEdge()` merupakan program untuk memanfaatkan seluruh layar.
6. Pada line [51] sampai line [59], `setContent{` untuk meneentukan isis dari UI yang menggunakan Jetpack Compose.
7. Pada line [52] sampai line [58], `TipCalculatorComposeTheme {` merupakan tema kustom aplikasi.
8. Pada line [53] sampai line [55], `Surface( modifier = Modifier)` merupakan program untuk menampilkan elemen latar belakang yang mengisi seluruh layar.
9. Pada line [56], TipCalculatorLayout() berfungsi untuk menampilkan layout utama kalkulator tip.
10. Pada line [65] sampai line [155], merupakan fungsi composable uatama untuk menampilkan UI kalkulator tip.
11. Pada line [66] sampai line [72], merupakan program untuk menyimpan input bill amount, menyimpan hasil perhitungan tip, menyimpan status pembulatan ke atas, menyimpan status apakah dropdown tip terbuka atau tidak, dan opsi persentase tip.
12. Pada line [74] sampai line [79], syntax merupakan membuat layout kolom yang scrollable dan rata tengah..
13. Pada line [81] sampai line [86], merupakan fungsi untuk menampilkan teks judul “Calculate Tip”

14. Pada line [88] sampai line [103], memanggil composable untuk input angka atau jumlah bill.
15. Pada line [105] sampai line [137], membuat dropdown untuk memilih tip percentage.
16. Pada line [109] sampai line [120], menampilkan textfield read-only sebagai anchor dropdown.
17. Pada line [122] sampai line [136], berfungsi untuk menampilkan pilihan dalam dropdown.
18. Pada line [139] sampai line [146], berfungsi untuk memanggil composable switch untuk mengaktifkan atau menonaktifkan pembulatan.
19. Pada line [148] sampai line [151], berfungsi untuk menampilkan hasil kalkulasi tip
20. Pada line [153], berfungsi untuk memberi jarak di bagian bawah.
21. Pada line [157] sampai line [161], fungsi pembantu untuk menghitung tip dari input string.
22. Pada line [163] sampai line [171], merupakan reusable composable untuk menginput angka dengan label dan icon.
23. Pada line [185] sampai line [190], menampilkan teks dan switch untuk pembulatan ke atas tip.
24. Pada line [209] sampai line [216], untuk menghitung besarnya tip dan mengubahnya ke format mata uang (USD)
25. Pada line [218] sampai line [223], menampilkan pratinjau layout utama di android studio preview.

SOAL 2

Jelaskan perbedaan dari implementasi XML dan Jetpack Compose beserta kelebihan dan kekurangan dari masing-masing implementasi.

D. Pembahasan

Perbedaan antara implemetasi XML dan Jetpack Compose adalah terletak pada struktur pemrogramannya. Pada XML pembuatan antarmuka atau tampilan aplikasi berada di file.xml, sedangkan untuk pembuatan logika program ada di file.kt yang berbasis Kotlin. Kelebihan dari XML ini adalah karena telah digunakan sejak lama XML lebih matang dalam syntax pemrograman dan sudah banyak didokumentasi. Kemudian, XMl juga akan lebih mudah dalam pengembangan aplikasi karena file terpisah antara program tampilan dan logika (jika tampilan tidak harus berubah secara real-time). Kekurangannya, sulit jika harus mengubah tampilan yang dinamis dan harus selalu berpindah-pindah antara file tampilan dan logika.

Sebaliknya, pada Jetpack Compose pembuatan program tampilan dan logika aplikasi tergabung di dalam satu file berbasis Kotlin. Kelebihannya adalah program lebih singkat, mudah digunakan jika memerlukan perubahan tampilan yang dinamis, danterintegrasi dengan fitur-fitur modern. Namun kekurangannya adalah karena masih baru jadi tidak semua library disupport, dan tidak semua fitur klasik android tersedia.

E. Tautan Git

Berikut adalah tautan untuk source code yang telah dibuat.
https://github.com/rizkiiirr/PemrogramanMobile.git